Applicant's Reply Brief filed March 19, 2007 has been noted.

In the virtual 3D space where the observer sees the 3D image corresponds to a 3D image display region; this display region is represented as **a viewing frustum consisting of 6 planes including the farthest and nearest depth planes** in which the observer sees the 3D image within the viewing frustum through the screen of the CRT and only an object within the viewing frustum appears on the screen.

The view frustum is defined (http://en.wikipedia.org/wiki/Viewing_frustum) as:

In 3D computer graphics, the **viewing frustum** or **view frustum** is the region of space in the modeled world that may appear on the screen; it is the field of view of the notional camera. The exact shape of this region varies depending on what kind of camera lens is being simulated, but typically it is a frustum of a rectangular pyramid (hence the name). The planes that cut the frustum perpendicular to the viewing direction are called the *near plane* and the *far plane*. Objects closer to the camera than the near plane or beyond the far plane are not drawn. Often, the far plane is placed infinitely far away from the camera so all objects within the frustum are drawn regardless of their distance from the camera.

Viewing frustum culling or **view frustum culling** is the process of removing objects that lie completely outside the viewing frustum from the rendering process. Rendering these objects would be a waste of time since they are not directly visible. In ray tracing, viewing frustum culling cannot be performed because objects outside the viewing frustum may be visible when reflected off an object inside the frustum. To make culling fast, it is usually done using bounding volumes surrounding the objects rather than the objects themselves.

Due the limit or finite capacity of the memory, the far plane must be set at some particular depth. In any convention Z-buffer display, the updates of the near plane and far plane serve the purpose of reduction of the processed data when the processed object is within a finite depth.

Delmlow's **viewing frustum** 714 is a box bounded within six planes including the far clipping plane (i.e., plane 715f). Delmlow specifically emphasizes that "the near and far clipping planes are positioned dynamically based on the part(s) (i.e., the cells that have non-null cell-to-art mapping) that are present within the five-point view frustum boundary 714 (column 12, lines 43-46)"; which means the updates of the close and far clipping planes 715e and 715f are based on the cells that have non-null cell-to-art mapping. More specifically, Delmlow states "the farthest point on the farthest cell (again with a non-null cell-to-part mapping) is used to position the far clipping plane" (column 12, lines 61-63); which means "whenever the farthest depth value does not match (i.e., nearer or farther) the current position of the far clipping plane, the far clipping plane will be updated to the farthest depth value;" or in a Z-pyramid as claimed "updating said far clipping plane based on the farthest depth value, if the farthest depth value is nearer than a depth of the far clipping plane."

Delmlow teaches the well known 3D data processing with Z-buffer using a viewing frustum which has the updatable z-near and z-far planes. The update for the Z-far plane is performed when the deepest point of the object is nearer than the current position of the z-far plane. The motivation of updating the z-far plane is reduction of depth data needed to be processed.

Greene teaches the well known of hierarchical Z-buffer for 3D data processing using a viewing frustum.

Simple substitution of the hierarchical z-buffer for the z-buffer in Delmlow's 3D processor will result a more efficient use of memory, but the update of z-far and z-near plane will work in the same principle for both of the regular z-buffer and the hierarchical z-buffer. Specifically, the z-far lane is updated to the farthest point of the object if the farthest point of the object is nearer than the current z-far plane. The motivation of updating the z-far plane in the hierarchical z-buffer is same as the motivation of updating the z-far plane in the z-buffer, in which the processing data is reduced by the reduction of the processed z-depth.

In Applicant's Reply Brief, Applicant repeats the same arguments as in the Appeal Brief filed May 10, 2006. See Examiner's answer sent January 17, 2007 for response to these arguments.

Applicant's Reply Brief filed March 19, 2007 has been noted.

Respecfully submitted,

/Phu K. Nguyen/

Primary Examiner, Art Unit 2628